JSON

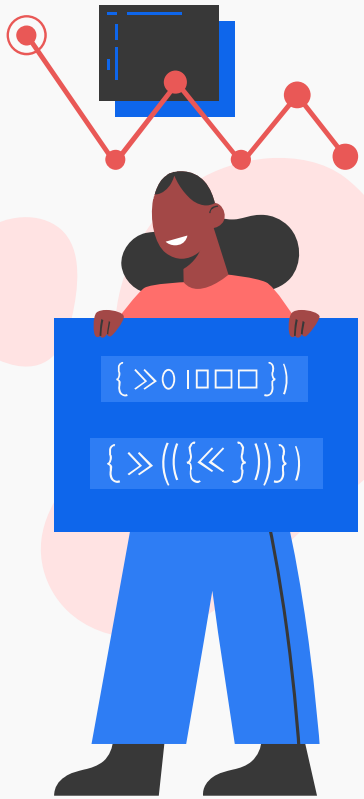
Desenv. org. interf. de usuário e elem visuais aplicados web

Msc. Lucas G. F. Alves
e-mail: LGFALVES@senacrs.com.br

Senac

Planejamento de Aula

- Revisão jQuery
- JSON
- Sintaxe
- Atividades



jQuery

O que é jQuery?

É um **framework** em javascript escrito por John Resig (nasceu em 2005).

Foi muito utilizada, leve com enorme número de plugins, porém conteúdo legado.

Estável com funcionalidades prontas para usar: **Ajax**.

CrossBrowser: funciona na maioria dos navegadores do mesmo jeito.

PS. Framework é uma biblioteca de javascript, ferramenta projetada com a intenção de facilitar o desenvolvimento de software.
PS2. Ajax é Asynchronous JavaScript and XML técnica de carregamento de conteúdo de página capaz se ser recuperada de um servidor.

jQuery

Como funciona?

Utiliza o DOM para navegar na árvore de elementos do HTML.

Manipula eventos javascript (Animações, efeitos, adiciona CSS).

Faz a mesma coisa que o DOM puro, no entanto, é mais prático: **“write less, do more”.**

Emprega o conceito de seletores, da mesma maneira que o CSS:

Pode-se utilizar seletores CSS3.

PS. DOM é Document Object Model -> são modelos de objetos expostos ao programador.

jQuery

Como utilizar?

Atualmente está na versão 3.7.1.

Disponível em https://jquery.com.

Pode ser baixado e importado diretamente do google.

```
<script src="jquery-3.7.1.min.js" ></script>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
```

Sintaxe

jQuery: usando seletores

```
//selecionando um campo do formulário
var form_campo = $("#id_do_meu_campo").click(funcion ());
var form_campo = jQuery("#id_do_meu_campo");
```

$ é um seletor do jQuery que seleciona os elementos, também se refere a uma "classe", ele instancia um objeto em jQuery.

.click() é invocado quando ocorre um clique do mouse no elemento selecionado.

```
//obtendo o valor deste campo
alert("form_campo.val()");
```

Sintaxe

jQuery: usando seletores

```
//selecionando um campo do formulário
var form_campo = document.getElementById(“#id_do_meu_campo”);
//obtendo o valor deste campo
alert(“form_campo.value”);

//código escrito em javascript puro para atribuir um valor.
document.getElementById(“#id_do_meu_campo”).value = 5
//mesmo código só que em jQuery.
$(‘#id_do_meu_campo’).val(5);
```

Sintaxe

jQuery: outros exemplos

```
//altera o css pelo jQuery
<script>
    $(document).ready(
        function(){
            $('h1').css({"background": "#cccccc"});
}
);
</script>
```

Sintaxe

jQuery: outros exemplos

```
//insere html passando como string
<script>
        $('<h1>Bem vindo ao Senac</h1>').prependTo('body');
</script>

//altera mais de um elemento ao mesmo tempo
<script>
        $('div,li').css({'background': '#fff000'});
</script>
```

Plugins

Animação

Animsition - A simple and easy jQuery plugin for CSS animated page transitions.
fakeLoader.js - Lightweight plugin that helps you create an animated spinner with a fullscreen loading mask to simulate the page preloading effect.
Fullpage.js - Create full screen pages fast and simple.
jQuery Transit - Super-smooth CSS3 transformations and transitions for jQuery.
Material Design Preloader!s - Recreation of the Material Design preloader.
Midnight - Switches fixed headers on the fly.
Parallax.js - Scrolling effect.
Scrollify - Assists scrolling and snaps to sections. Touch optimised.
Waves - Click effect inspired by Google's Material Design.
jQuery DrawSVG - Lightweight, simple to use jQuery plugin to animate SVG paths
jQuery Particles - A plugin to easily add Particles animations to your web application

Plugins

Forms

Bootstrap Multiselect - Multiselect for Bootstrap.
File Upload - File Upload widget with multiple file selection, drag&drop support, progress bar, validation and preview images, audio and video.
Ideal Forms - Framework for building and validating responsive HTML5 forms.
jQuery Form Plugin - Easy and unobtrusive HTML forms upgrade to use AJAX.
jquery-minicolors - A tiny color picker plugin.
Justified Gallery - Allows you to create a gallery with a justified grid.
Labelauty - A lightweight and beautiful plugin for radio and checkbox inputs.
Payform - A library (with jQuery plugin) for building credit card forms, validating inputs, and formatting numbers.
Pickadate - The mobile-friendly, responsive, and lightweight date & time input picker.
Select2 - Select box with support for searching, tagging, remote data sets, infinite scrolling.
selectize.js - jQuery based hybrid of a textbox and <select> box.

Plugins

Images, Maps and Charts

AnyChart-jQuery - Plugin for easily using AnyChart JavaScript charting library with jQuery.
Chart.js - Simple HTML5 Charts using the <canvas> tag.
Gridder - Displays a thumbnail grid expanding preview similar to the effect seen on Google Images.
jquery.sparkline - Generate small sparkline charts.
jQuery Mapael - Plugin based on raphael.js that allows you to display dynamic vector maps.
jQueryGantt - Gantt editor.
Nivo Slider - Beautiful and easy to use image slider.
Owl Carousel 2 - Responsive carousel slider.
Slick - The last carousel you'll ever need.
Peity - Progressive <svg> pie, donut, bar and line charts.
Unite Gallery - Responsive jQuery image and video gallery plugin.
Viewer - A simple jQuery image viewing plugin.
JQuery-linechart - Simple and lightweight library for creating line charts

Plugins

Menus

jQuery contextMenu - Management facility for context menus.
jPanelMenu - Creates a paneled-style menu (like the type seen in the mobile versions of Facebook, Google and native iPhone applications).
jQuery-menu-aim - Fires events when user's cursor aims at particular dropdown menu items.
mmenu - App look-alike on- and off-canvas menus with sliding submenus.
Multi-level push menu - Allows endless nesting of navigation elements.
Slidebars - jQuery framework for off-canvas menus and sidebars.
stickUp - Sticks an element to the top of the browser window while scrolling past it, always keeping it in view.
Superfish - Adds usability enhancements to existing multi-level drop-down menus.
Yamm - Yet another megamenu for Bootstrap 3.

Dinâmica com jQuery

1) Criar uma página da web que use **jQuery** para criar um **carrossel de imagens**. Plugin de referencia - **Owl Carousel 2**. Idéia de uso: aplicar no blog.

2) Criar uma página aplicando um plugin jQuery com efeito **parallax**. Idéia de uso: Aplicar a página de contato.

Desafio: Criar outra página e aplicar qualquer outro plugin de jquery.

Fundo de tela: **Poligonizer**.

Validação de formulários: **jQueryValidation**

({(({≫})) ≪ }

JSON

JSON

JSON a sigla é derivada de **JavaScript Object Notation**.

Utilizado em interfaces baseadas em HTML para **armazenar dados em memória**.

Leve para **envio/recebimento de informações de serviços remotos**.

Os dados definidos em JSON são definidos no mesmo **formato que objetos JavaScript**, portanto, é de fácil entendimento e manipulação.

Mais simples que um XML que possam prejudicar ou dificultar a leitura das propriedades.

Também não requer nenhum parser sofisticado para converter uma estrutura em variáveis JavaScript.

JSON

Ao enviar dados com este formato é necessário formatá-los em uma string.

Na **leitura/recebimento**, é necessário **converter a string** no padrão da notação em uma estrutura **JavaScript**.

Por outro lado, em casos de dados estruturação por tag, como o **XML**, é necessário um **parser** mais sofisticado para converter a **string numa estrutura DOM** (Document Object Model), interpretar nodos da estrutura e ainda extrair os valores da estrutura para armazená-los em variáveis JavaScript, portanto, requer mais esforço.

JSON

Sintaxe

// É comum aplicações JS invocarem serviços remotos e receber uma string JSON
// como resposta. JSON.parse converte esta string em **objeto JavaScript**.
var respostaServer = JSON.parse(responseText);

Arquivos JSON possuem extensão "**.json**" e o MIME type (tipo de mídia) definido para texto JSON é "**application/json**".

Em resumo é mais rápido, mais fácil, mais leve que um XML e produtivo trabalhar com JSON.

JSON

Definição de propriedades

Uma propriedade é composta de duas partes: a chave e o valor.

Exemplos:

- Strings - “Olá mundo”;
- Números - 1 ou 56.32;
- Arrays - [1, 2, 3] ;
- Objetos - {“nome”:”Fulano”};
- Dados nulos - null;

No JSON é necessário colocar **nome de atributo entre aspas-duplas**.

Exemplo: "nome" : "Fulano"

JSON

Definição de Objetos de Javascript

Objetos são estruturas que mantém um conjunto de propriedades sobre um mesmo indivíduo, ou simplesmente objeto.

Por exemplo, de uma **pessoa**, podemos manter atributos de nome, idade e local de residência. Este conjunto de informações relacionadas são as propriedades de um objeto.

Para definir um objeto em **JSON** devemos fazer uso de caracteres de { } (chaves).

As chaves que definem um objeto devem envolver as propriedades deste objeto.

JSON

Definição de Objetos

Exemplo:

```
{ //abertura do formato json com {
        "nome" : "Fulano",  //chave e valor separado por :
        "idade" : 30,  //cada propriedade é separado por vírgula
        "local" :      {
                          "pais" : "Brasil",
                          "uf" : "RS",
                          "cidade" : "Porto Alegre"
                        }
 } //fechamento do arquivo com }
```

JSON

Definição de Arrays de Objetos

A definição de um array é feita com os caracteres [ ].

```
{       "pessoas" : [       { //primero objeto pessoa
                                "nome" : "Fulano",
                                "idade" : 30,
                                "local" :       {
                                                "pais" : "Brasil",
                                                "uf" : "RS",
                                        }
                        }, //separa cada objeto com vírgula
                        { //segundo objeto pessoa
                                "nome" : "Beltrano",
                        }] }
```

JSON

Exemplo de Javascript

```
// objeto JSON (um array) armazenado em objeto JavaScript
var pessoas =
// Lógica de array com []
[
    // definição de um objeto com {}
    {
      // proprieades ou atributos são chave-valor
      "nome":"Fulano",
      "idade": 30,
      // é possível associar um outro objeto como um valor de propriedade
      "local":{
         "pais":"Brasil",
         "uf":"RS",
         "cidade":"Porto Alegre"
      }
    },
    {
      "nome":"Beltrano",
      "idade": 32,
      "local":{
         "pais":"Brasil",
         "uf":"RS",
         "cidade":"Montenegro"
      }
    }
];
document.getElementById("demo").innerHTML =
"Primeiro registro do array contém dados do <b>\"" + pessoas[0].nome + "\"</b> de <b>" + pessoas[0].local.cidade + "</b>";
```

JSON

Leitura em arquivos JSON

Para carregar um arquivo JSON é necessário utilizar a função fetch. Após é chamado a função response para verificar se o arquivo retornou algo. Se retornou é salvo o dado em uma variável que ao percorrê-la terá os dados do arquivo JSON.

```
Exemplo: fetch("./JSON/data.json")
            .then(response => response.json())
            .then(dados => {
                // Itera sobre os dados JSON
                for (const curso in dados) {
```

JSON

Escrita em arquivos JSON

Para carregar um arquivo JSON é necessário utilizar a função fetch. Após é chamado a função response para verificar se o arquivo retornou algo. Se retornou é salvo o dado em uma variável que ao percorrê-la terá os dados do arquivo JSON.

```
Exemplo: fetch("./JSON/data.json")
            .then(response => response.json())
            .then(dados => {
                // Itera sobre os dados JSON
                for (const curso in dados) {
```

Exercício

Exercício

1) Criar os arquivos html, css, js e um JSON contendo 5 nomes de objetos. Os 5 nomes dos objetos devem ser mostrados no html.

2) Criar outro arquivo JSON com 5 objetos de pessoas com seus respectivos dados como nome, idade, cpf, telefone (pelo menos 5 usuários) e imprima os usuários cadastrados.

3) Criar um programa que leia um arquivo JSON contendo os alunos e seus respectivos cursos (pelo menos 5 cursos) e imprima os alunos matriculados em cada curso. Utilizem a função Fetch para ler o arquivo.

Exemplo JSON:

```
{
  "Matemática": [
    "João", "Maria", "Pedro"
  ],
  "Português": [
    "Fulano", "Beltrano", "Ciclano"
  ]
```

Professor

Lucas G. F. Alves

Senac

Obrigado!

E-mail :LGFALVES@senacrs.com.br